

Migration Access zu SQL Server

AEK 11 – Nürnberg, Oktober 2008

Bernd Jungbluth

www.berndjungbluth.de



Migration Access zu SQL Server

Vorstellung

- Bernd Jungbluth

- IT-Erfahrung
 - SQL und Datenbanken seit 1991
 - Access seit Version 2.0
 - SQL Server seit Version 7.0

- Freiberuflicher Berater und Entwickler
 - Administration, Entwicklung, Optimierung von und mit SQL Server
 - Migration von Access nach SQL Server



Migration Access zu SQL Server

Agenda

- Die Geschichte einer Access-Datenbank
- Die Gründe für eine Migration
- Die Migration
- Die neue Datenbank und Access
- Die Optimierung



Migration Access zu SQL Server

Die Geschichte einer Access-Datenbank – Teil 1

- Am Anfang steht eine Informationslücke.
- „Wir brauchen eine Datenbank!“
- Auch kleine Datenbanken werden groß.
 - Mehr Benutzer
 - Mehr Daten
- Die Datenbank ist für das Unternehmen wichtig!
 - Steigender Informationsbedarf
 - Steigender Informationsgehalt



Migration Access zu SQL Server

Die Geschichte einer Access-Datenbank – Teil 2

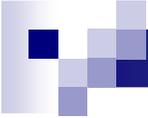
- Die Datenbank wird langsam und instabil.
- Gründe:
 - Access ist eine Desktop-Datenbank.
 - „Eine Datenbank für einen Benutzer“.
- Lösung:
 - Die Datenbank wird in Frontend und Backend aufgeteilt.
 - Das Backend wird über ein Netzwerklaufwerk bereitgestellt.
- Ergebnis:
 - Als Lösung erhält man eine Datenbank im Dateiserver-Betrieb.
 - Die Datenbank wird langsam und instabil.



Migration Access zu SQL Server

Die Gründe für eine Migration

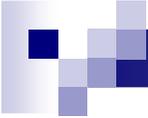
- Performance
- Sicherheit
 - Stabilität der Daten
 - Ausfallsicherheit
 - Zugriffssicherheit
- Skalierbarkeit
- Flexibilität



Migration Access zu SQL Server

Gründe – Performance

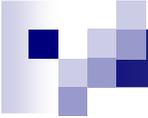
- Access-Frontend mit Access-Backend
 - Mehrere Benutzer arbeiten gleichzeitig mit ein und derselben Datenbank-Datei.
 - Jeder Benutzer verarbeitet die benötigten Daten über das Netzwerk direkt in der Datenbank-Datei.
 - Zur Verarbeitung werden alle Daten der benötigten Tabellen als Kopie an die jeweiligen Clients übertragen.
- Nachteile:
 - Hohe Netzwerklast
 - Keine Nutzung der Server-Ressourcen
 - Datei-Server-Betrieb



Migration Access zu SQL Server

Gründe – Performance

- Access-Frontend mit SQL Server-Datenbank
 - Die Anforderungen zur Datenverarbeitung werden an den SQL Server-Dienst übergeben.
 - Der SQL Server-Dienst verarbeitet die benötigten Daten und überträgt das Ergebnis an den Client.
- Vorteile:
 - Minimierung der Netzwerklast
 - Nutzung der Server-Ressourcen
 - Client-/Server-Betrieb



Migration Access zu SQL Server

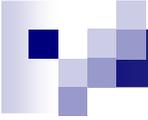
Gründe – Sicherheit – Stabilität der Daten

■ Access

- Fehlerhafte Netzwerkübertragungen können zu Dateninkonsistenz führen.
- Ein Wiederherstellen der letzten Aktion ist nicht möglich.
- Die Datenbank kann nur repariert und komprimiert werden.

■ SQL Server

- Fehlerhafte Netzwerkübertragungen können keine Dateninkonsistenz verursachen.
- Jede Aktion wird im Transaktionsprotokoll aufgezeichnet und erst nach der Ausführung abgeschlossen.
- Ein Wiederherstellen der letzten Aktion ist möglich.



Migration Access zu SQL Server

Gründe – Sicherheit – Ausfallsicherheit

■ Access

- Datensicherung per externe Sicherungssoftware
- Keine Datensicherung im laufenden Betrieb möglich
- Wiederherstellung mit letzter Sicherung

■ SQL Server

- Datensicherung mit eigenen Sicherungsfunktionen zur Vollsicherung, differentiellen und Transaktionsprotokollsicherung
- Datensicherung im laufenden Betrieb möglich
- Wiederherstellung bis zur letzten Transaktionsprotokollsicherung
- Weitere Ausfallsicherheit durch Datenbankspiegelung, Failover-Cluster oder Protokollversand



Migration Access zu SQL Server

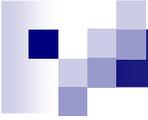
Gründe – Sicherheit – Zugriffssicherheit

■ Access

- Datenbank-Kennwort
- Arbeitsgruppen-Informationsdatei (MDW)

■ SQL Server

- Mehrstufige Sicherheitsarchitektur
- SQL Server- und Windows-Authentifizierung (Login)
- Datenbank-Benutzer (User)
- Dedizierte Rechtevergabe an Objekten bis auf Spaltenebene
- Datenverschlüsselung bei Übertragung und in der Datenbank



Migration Access zu SQL Server

Gründe – Skalierbarkeit

- Grenzen von Access:
 - Maximal 2 GigaByte Daten
 - Maximal 255 Benutzer
 - Ertragbare Anzahl bei 20 Benutzern

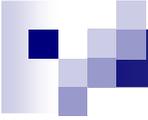
- Grenzen von SQL Server:
 - Maximal 5 GigaByte Daten pro Datenbank bei Express Edition
 - Maximal 1 PetaByte pro Datenbank ab Standard Edition
 - Maximal 32.767 Benutzerverbindungen



Migration Access zu SQL Server

Gründe – Flexibilität

- Logik zur Datenverarbeitung in Access
 - Abfragen, Formulare, Berichte, Module und Makros
 - Logik nur im Access-Frontend verfügbar
 - Mehrfache Nutzung der Logik in anderen Frontends nur durch Neuentwicklung möglich
- Logik zur Datenverarbeitung in SQL Server
 - Sichten, Gespeicherte Prozeduren, Trigger und Benutzerdefinierte Funktionen
 - Logik für unterschiedliche Frontends verfügbar
 - Mehrfache Nutzung der Logik in anderen Frontends durch Verwendung der SQL Server-Objekte möglich



Migration Access zu SQL Server

Die Migration

- Migration planen
- Möglichkeiten prüfen und testen
 - Manuelle Migration
 - Upsizing-Assistent
 - SQL Server 2005 Migrations-Assistent für Access
- Datenbank sichern
- Test-Migration durchführen
- Migrierte Datenbank prüfen
 - Tabellen mitsamt den Daten
 - Abfrageverhalten der Access-Applikation



Migration Access zu SQL Server

Migration – Planung

■ Neuentwicklung

- Chance zur Beseitigung alter Fehler und Altlasten
- Kosten/Nutzen prüfen
- Entwicklungszeit einplanen
- Schulung für Entwickler und Anwender einplanen

■ Migrieren

- .NET
- ADP mit direktem Zugriff auf SQL Server
- MDB mit eingebundenen Tabellen
- ACCDB mit eingebundenen Tabellen

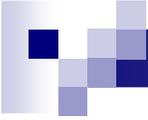


Migration Access zu SQL Server

Migration – Manuelle Migration

- Manuelle Migration
 - Bei überschaubarer Anzahl von Tabellen und Abfragen
 - Entwicklung im SQL Server Management Studio

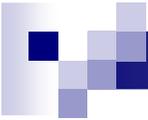
- Vorgehensweise:
 - Tabellen anlegen
 - Primärschlüssel und Indizes definieren
 - Fremdschlüssel und referentielle Integrität definieren
 - Standardwerte und Einschränkungen konfigurieren
 - Per Import-/Export-Assistent Daten importieren



Migration Access zu SQL Server

Migration – Upsizing Assistent

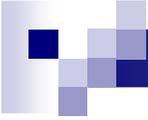
- Upsizing Assistent
 - Seit Access 97 verfügbar
 - Seit Access 2000 in Access integriert
- Migrationsvarianten:
 - Nur SQL Server-Datenbank
 - MDB mit eingebundenen SQL Server-Tabellen
 - ADP inkl. Migration der Access-Abfragen
- Eingeschränkte Konfigurationsmöglichkeiten
- Ergebnisbericht



Migration Access zu SQL Server

Migration – SSMA-A

- SQL Server 2005 Migrations-Assistent für Access
 - Seit SQL Server 2005 als kostenfreier Download verfügbar
 - Notwendige, aber kostenfreie Lizenzierung
- Migrationsvarianten:
 - Migration von Tabellen und Abfragen
 - MDB mit eingebundenen SQL Server-Tabellen
- Konfigurationsmöglichkeiten:
 - Dedizierte Auswahl von Tabellen und Abfragen
 - Definition der Datentypen von Access nach SQL Server
- Aufwandskalkulation und Ergebnisbericht



Migration Access zu SQL Server

Migration – SSMA-A

- Simulation der Migration
 - Ausgabe der Migrationsergebnisse in Berichten
 - Gezielte Analyse der Warnungen und Fehler
- Dreistufige Migration
 - Erstellen eines Skripts zur Anlage der Tabellen und Abfragen
 - Ausführen des Skripts in der SQL Server-Datenbank
 - Exportieren der Daten von Access nach SQL Server
- Migration als Projekt
 - Ausführen und Speichern von Teilschritten
 - Fortsetzung der Migration zu späterem Zeitpunkt möglich



Migration Access zu SQL Server

Die neue Datenbank und Access

- Prüfen der neuen Datenbank
 - Tabellen
 - Daten
 - Einschränkungen und Standardwerte
 - Primärschlüssel und Indizes
 - Fremdschlüssel und referentielle Integrität
- Access-Datentypen und SQL Server-Datentypen
 - Unterschiedliche Typen
 - Unterschiedliche Genauigkeiten



Migration Access zu SQL Server

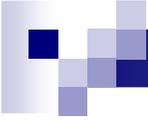
Die neue Datenbank und Access

■ Einschränkungen

- Aus Standardwerten werden Defaults.
- Aus Gültigkeitsregeln werden Check Constraints.
- Aus „Leere Zeichenfolge“ wird ein Check Constraint.
- Aus „Eingabe erforderlich“ wird „NOT NULL“.

■ Primärschlüssel

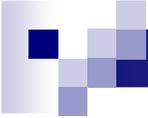
- Der Upsizing-Assistent legt den Primärschlüssel als „nicht gruppierter Index“ an.
- Eine Tabelle sollte einen „gruppierten Index“ enthalten.



Migration Access zu SQL Server

Exkurs – Gruppiertes und nicht gruppiertes Index

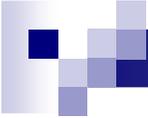
- Gruppiertes Index (clustered Index)
 - „Kein“ Index
 - Physikalische Sortierung der Tabelle
 - Im Vergleich mit einem Buch: das Buch selbst
 - Nur ein gruppiertes Index pro Tabelle möglich
- Nicht gruppiertes Index (nonclustered Index)
 - Ein Index
 - Index-Seiten mit Verweisen auf die jeweiligen Daten-Seiten
 - Im Vergleich mit einem Buch: das Stichwortverzeichnis
 - Mehrere nicht gruppierte Indizes pro Tabelle möglich



Migration Access zu SQL Server

Die neue Datenbank und Access

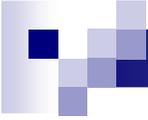
Access-Datentyp	SQL Server-Datentyp
Autowert (Long Integer)	integer (mit Option Identity)
Autowert (Replikations-ID)	uniqueidentifier (mit Default NewId())
Zahl (Byte)	smallint
Zahl (Integer)	smallint
Zahl (Long Integer)	integer
Zahl (Single)	real
Zahl (Double)	float
Zahl (Dezimal)	decimal(18,0)
Zahl (Replikations-ID)	uniqueidentifier
Währung	money



Migration Access zu SQL Server

Die neue Datenbank und Access

Access-Datentyp	SQL Server-Datentyp
Ja/Nein	bit (mit Option NOT NULL)
Text	nvarchar(n)
Memo	ntext
Hyperlink	ntext (ohne Hyperlink-Funktionalität)
Datum/Uhrzeit	datetime
OLE-Objekt	image
Mehrwertige Spalten (neu in Access 2007)	ntext
Anlage (neu in Access 2007)	ntext
Text als RichText (neu in Access 2007)	ntext (inklusive HTML-Tags für Formatierung)



Migration Access zu SQL Server

Die neue Datenbank und Access

- Access und SQL Server-Tabellen
 - SQL Server-Tabellen in Access-Applikation einbinden
 - Per Tabellenverknüpfungsmanager oder VBA
 - ODBC-Verbindung notwendig
- Jet-Engine und eingebundene Tabellen
 - Verwaltung der SQL Server-Tabellen durch Jet-Engine
 - Assimilation der SQL Server-Tabellen als Access-Tabellen inklusive lokaler Verwendung von Access-Datentypen
 - Keine Strukturänderung der Tabelle innerhalb Access möglich
 - Keine automatische Aktualisierung von Strukturänderungen der Tabellen



Migration Access zu SQL Server

Die neue Datenbank und Access

- Demo

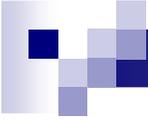


Migration Access zu SQL Server

Die Optimierung

- Access und Datenbank
 - Tabellen
 - Abfragen

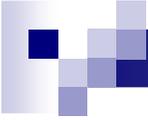
- Access und Applikation
 - Formulare
 - Berichte
 - VBA



Migration Access zu SQL Server

Optimierung – Tabellen

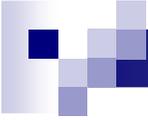
- Tabellen ohne Primärschlüssel sind Snapshots.
- Die Eigenheiten eines Snapshots:
 - Alle Datensätze mitsamt allen Spalten werden eingelesen.
 - Die eingelesenen Daten werden nicht automatisch aktualisiert.
 - Die Daten sind schreibgeschützt.
- Snapshots erzeugen zwar wenige Netzwerkzugriffe, dafür aber eine hohe Netzwerklast, da alle Datensätze einer Tabelle übertragen werden.



Migration Access zu SQL Server

Optimierung – Tabellen

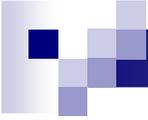
- Tabellen mit Primärschlüssel sind Dynasets.
- Die Eigenheiten eines Dynasets:
 - Nur die Primärschlüssel der Datensätze werden eingelesen.
 - Die Datenspalten werden nur für die angezeigten Datensätze zzgl. der 10 vorangegangenen und der 10 folgenden eingelesen.
 - Die eingelesenen Daten werden alle 60 Sekunden aktualisiert.
 - Die Daten sind nicht schreibgeschützt.
- Dynasets erzeugen zwar mehr Netzwerkzugriffe, dafür aber weniger Netzwerklast, da nur wenige Daten übertragen werden.



Migration Access zu SQL Server

Optimierung – Tabellen

- Primärschlüssel und eingebundene Tabellen
 - Die Jet-Engine bestimmt den Primärschlüssel der Tabelle.
 - Der Primärschlüssel der SQL Server-Tabelle wird ignoriert.
 - Die erste Wahl ist der gruppierte Index der Tabelle.
 - Die zweite Wahl ist der erste alphanumerisch sortierte Index.
 - Die Auswahl sollte unbedingt geprüft werden.
- Manuelle Definition des Primärschlüssels
 - Der gruppierte Index ist hier nicht die erste Wahl.
 - Integer-Spalten sind alphanumerischen Spalten vorzuziehen.
 - Spalten mit Gleitkommazahlen liefern „#Fehler“ anstatt Daten.



Migration Access zu SQL Server

Optimierung – Tabellen

- **Bessere Performance durch timestamp-Spalte**
 - Tabellen mit Gleitkomma- und Binärspalten sollten mit einer timestamp-Spalte erweitert werden.
 - timestamp ist kein Zeitstempel, sondern ein Binärwert.
 - timestamp kennzeichnet den Zustand des Datensatzes.
- **Schnellere Ermittlung geänderter Datensätze**
 - Ohne eine timestamp-Spalte wird beim Speichern eines Datensatzes jeder einzelne Wert der Access-Spalte mit dem zugehörigen Wert der SQL Server-Spalte verglichen.
 - Mit einer timestamp-Spalte werden nur die Werte der timestamp-Spalten verglichen.



Migration Access zu SQL Server

Optimierung – Tabellen

- Demo



Migration Access zu SQL Server

Werbung

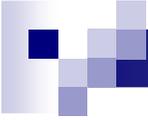
- Workshop „Migration Access zu SQL Server“
 - 1 Tag
 - Maximal 7 Teilnehmer
- Inhalt:
 - Vertiefung dieses Themas
 - Beispiele und Übungen
- Mögliche Orte:
 - Hotel Birkenhof im Hunsrück
 - Vor Ort
 - Diverse Schulungszentren



Migration Access zu SQL Server

Optimierung – Abfragen

- Der Performancekiller schlechthin!
- Betrifft primär Access-Abfragen mit:
 - Ausdrücken bzw. Funktionen, Verweisen auf Formularfelder
 - Komplexen Tabellenverknüpfungen, Unterabfragen
 - TOP-Anweisungen, berechneten Spalten u.v.m.
- Access-Abfragen mit SQL Server-Tabellen
 - Der Access-Abfrageoptimierer erstellt einen Ausführungsplan.
 - Anhand des Ausführungsplans wird entschieden in welcher Form die Abfrage an den SQL Server übergeben wird.
 - Die Entscheidung kann nicht beeinflusst werden.



Migration Access zu SQL Server

Optimierung – Abfragen

- Übergabe der kompletten Abfrage
 - Die Abfrage wird auf dem SQL Server ausgeführt.
 - Es wird nur die Ergebnismenge an den Client übertragen.
- Übergabe der Abfrage in mehreren einzelnen Abfragen
 - Die Abfrage wird in mehrere Abfragen gesplittet.
 - Es werden Teilergebnismengen an den Client übertragen.
 - Die ursprüngliche Abfrage wird lokal ausgeführt.
- Keine Übergabe der Abfrage
 - Es werden alle Daten der in der Abfrage enthaltenen Tabellen an den Client übertragen.
 - Die Abfrage wird lokal ausgeführt.



Migration Access zu SQL Server

Optimierung – Abfragen

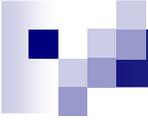
- Ermitteln der Performancesünder
 - Manuelle Suche nach den bekannten Kriterien
 - ODBC-Konfiguration „Abfragen mit langer Laufzeit speichern“
 - SQL Server Profiler
- SQL Server Profiler
 - Aufzeichnung aller vom SQL Server ausgeführten SQL-Anweisungen in Trace-Datei oder Tabelle
 - Umfangreiche Konfiguration der Aufzeichnung nach Überwachungstypen und Attributen
 - Filterung der Aufzeichnung nach verschiedenen Kriterien



Migration Access zu SQL Server

Optimierung – Abfragen

- Demo



Migration Access zu SQL Server

Optimierung – Abfragen

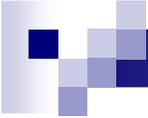
- Optimierung der Abfragen
 - Migration der Abfragen nach SQL Server
 - Ablösen durch Sichten und Gespeicherte Prozeduren
- Migration:
 - Upsizing-Assistent mit Migrationstyp ADP
 - Manuell mit SQL Server Management Studio
- Vorteile:
 - Bessere Auslastung der Hardware
 - Bessere Performance durch Abfrage- bzw. Prozedurcache



Migration Access zu SQL Server

Exkurs – Abfrage- bzw. Prozedurcache

- Abfrage- bzw. Prozedurcache
 - Reservierter Bereich im Arbeitsspeicher
 - Speicherort für Ausführungspläne von SQL Server-Objekten
- Anwendung des Ausführungsplans
 - Erstellen des Ausführungsplans und Speichern im Cache beim erstmaligen Aufruf des SQL Server-Objekts
 - Wiederverwendung des Ausführungsplans ab dem zweiten Aufruf
- Inhalt eines Ausführungsplans
 - Reihenfolge der Zugriffe auf Tabellen
 - Zugriff auf Indizes
 - Sortierungen und Aggregationen u.m.



Migration Access zu SQL Server

Optimierung – Abfragen

- Access-Abfragen zu SQL Server-Objekten

Access-Abfrage	SQL Server-Objekt
Auswahlabfrage	Sicht oder Gespeicherte Prozedur
Anfügeabfrage	Gespeicherte Prozedur mit Insert
Löschabfrage	Gespeicherte Prozedur mit Delete
Aktualisierungsabfrage	Gespeicherte Prozedur mit Update
Parameterabfrage	Gespeicherte Prozedur
Tabellenerstellungsabfrage	Gespeicherte Prozedur mit Select Into
Union	Sicht oder Gespeicherte Prozedur
Kreuztabellenabfrage	Sicht oder Gespeicherte Prozedur



Migration Access zu SQL Server

Exkurs – SQL Server Objekte

- Sichten
- Gespeicherte Prozeduren

- Benutzerdefinierte Funktionen
 - Skalarfunktionen
 - Tabellenfunktionen
 - Inlinefunktionen
- Trigger



Migration Access zu SQL Server

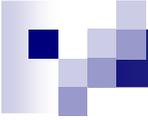
Exkurs – Sichten

■ Sichten

- Vergleichbar mit Access-Auswahlabfragen
- Liefern Zeilen und Spalten von einer oder mehreren Tabellen
- Keine Parameterübergabe möglich

■ Anwendung von Sichten:

- In Sichten
- In Gespeicherten Prozeduren
- In Select-Anweisungen
- In Benutzerdefinierten Funktionen
- In Triggern



Migration Access zu SQL Server

Optimierung – Abfragen

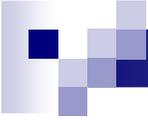
- Access und Sichten
 - Sichten als Tabellen in die Access-Applikation einbinden
 - Per Tabellenverknüpfungsmanager oder VBA
 - ODBC-Verbindung notwendig
 - Definition eines Primärschlüssels möglich
- Jet-Engine und eingebundene Sichten
 - Verwaltung der Sichten durch Jet-Engine
 - Assimilation der Sichten als Access-Tabellen
 - Keine Strukturänderung der Sicht innerhalb Access möglich
 - Keine automatische Aktualisierung bei Strukturänderungen der Sichten



Migration Access zu SQL Server

Optimierung – Abfragen

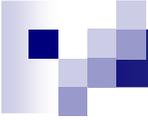
- Demo



Migration Access zu SQL Server

Exkurs – Gespeicherte Prozeduren

- Gespeicherte Prozeduren
 - Ähneln im weitesten Sinne einer VBA-Funktion
 - Geeignet für Datenaufbereitung
 - Einziges SQL Server-Objekt zur Datenmanipulation
 - Programmierung in T-SQL
- Anwendung von Gespeicherten Prozeduren:
 - Manueller Aufruf per Execute
 - In Gespeicherten Prozeduren
 - In Triggern
 - Nicht in Sichten, Select-Anweisungen oder Benutzerdefinierten Funktionen anwendbar



Migration Access zu SQL Server

Exkurs – T-SQL

- Programmiersprache des SQL Servers
- Beste Alternative zur Datenverarbeitung in SQL Server
- Einige Möglichkeiten in T-SQL:
 - Eingabe- und Ausgabeparameter nutzen
 - Variablen, Temporäre Tabellen und Table-Variablen für Zwischenergebnisse verwenden
 - IF-ELSE-Anweisung und Schleifen programmieren
 - Systemwerte abfragen
 - Sichten, Gespeicherte Prozeduren und Benutzerdefinierte Funktionen verwenden



Migration Access zu SQL Server

Exkurs – Gespeicherte Prozeduren

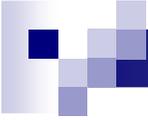
- Optimal zur Einhaltung von Geschäftsregeln und Gewährleistung der Datenkonsistenz
 - Einheitliche und gekapselte Verarbeitung der Daten
 - Änderung von Regeln nur in den betroffenen Gespeicherten Prozeduren notwendig, nicht in mehreren Frontends
 - Anpassung im Frontend nur notwendig bei geänderter Parameterübergabe der Gespeicherten Prozedur
- Vorgehensweise:
 - Datenverarbeitung nur über Gespeicherte Prozeduren
 - Direkte Datenverarbeitung an Tabellen verweigern



Migration Access zu SQL Server

Optimierung – Abfragen

- Access und gespeicherte Prozeduren
 - Aufruf per Pass Through-Abfragen
 - ODBC-Verbindung notwendig
- Pass Through-Abfragen
 - Direkter Durchgriff auf den SQL Server (das Backend)
 - Keine „Optimierung“ durch den Access-Abfrageoptimierer
 - Programmierung in T-SQL (Sprache des Backend)
 - Aufruf einfacher SQL-Anweisungen möglich
- Pass Through-Abfragen sollten immer in Verbindung mit Gespeicherten Prozeduren verwendet werden!



Migration Access zu SQL Server

Optimierung – Abfragen

- Pass Through-Abfrage zur Datenaufbereitung
 - Als Datenherkunft für Berichte, Formulare und Controls
 - Als Abfrage in Access-Abfragen
- Pass Through-Abfrage zur Datenmanipulation
 - Bessere Performance als Access-Abfragen
 - Ausführung einer einzelnen Anweisung auf SQL Server
- Pass Through-Abfragen und Parameter
 - Keine direkte Parameterübergabe möglich
 - „Übergabe“ nur per VBA durch Änderung des Quellcodes der Pass Through-Abfrage möglich



Migration Access zu SQL Server

Optimierung – Abfragen

- Demo



Migration Access zu SQL Server

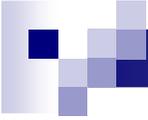
Optimierung

- Datenbank und Access

- Tabellen
- Abfragen

- Access-Applikation

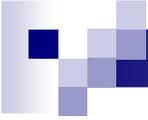
- Formulare
- Berichte
- VBA



Migration Access zu SQL Server

Optimierung – Formulare und Berichte

- Performancekiller Informationsüberfluss
- Datenanzeige einschränken
 - Nur primäre Informationen direkt anzeigen
 - Sekundäre Informationen auf Registerkarten verteilen
 - Menge der Daten per Auswahldialog vorab filtern lassen
- Datenübertragung einschränken
 - Daten beim Einlesen so gut es geht filtern
 - Nur Daten der sichtbaren Anzeige laden
 - Daten diverser Controls (z.B. Registerkarten) erst bei deren Aktivierung einlesen



Migration Access zu SQL Server

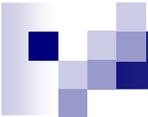
Optimierung – Formulare und Berichte

■ Datenherkunft

- Pass Through-Abfragen mit Gespeicherten Prozeduren
- Select-Anweisungen mit Where-Bedingung
- Keine Tabellen oder Access-Abfragen

■ Kombo- und Listboxen

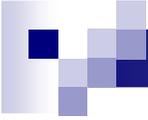
- Nur zur Auswahl von Datensätzen verwenden
- Nicht zur Anzeige von weiteren Ausgabefeldern verwenden
- Daten bei großen Datenmengen erst bei Aktivierung einlesen
- Daten beim Einlesen so gut es geht filtern



Migration Access zu SQL Server

Optimierung – VBA

- Schleifen so gut es geht vermeiden
- Definition von Recordsets
 - Daten beim Einlesen so gut es geht filtern
 - Select-Anweisungen mit Where-Bedingung oder Pass Through-Abfragen als Datenherkunft verwenden
 - Zugriffsart auf die Anforderung anpassen
- Datenmanipulation mit Recordsets
 - Neue Datensätze mit Gespeicherter Prozedur anlegen
 - Datensätze mit Gespeicherter Prozedur ändern oder löschen
 - Auch in Schleifen Gespeicherte Prozeduren verwenden



Migration Access zu SQL Server

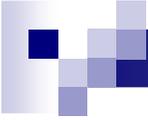
Zusammenfassung

■ Migration

- Aus Gründen der Performance, Sicherheit, Skalierbarkeit und Flexibilität sinnvoll
- Unterstützung durch den Upsizing-Assistent oder SSMA-A
- Überprüfung der neuen Datenbank auf Struktur und Daten

■ Optimierung

- Kontrollierte Tabellenverknüpfung beim Start der Applikation
- Ablösen der Access-Abfragen durch SQL Server-Objekte
- Anpassen der Anwendungslogik an die neuen Objekte



Migration Access zu SQL Server

In Zukunft zu beachten

- Der „Client/Server-Gedanke“
 - Der SQL Server ermittelt die Daten.
 - Der SQL Server manipuliert die Daten.
 - Der Client formatiert die Daten und zeigt diese an.
 - Es werden nur die notwendigen Datensätze eingelesen.
 - Es gibt noch andere Benutzer.
- Der „SQL Server-Gedanke“
 - SQL ist Mengenlehre und Mengenverarbeitung.
 - Die Ausführungspläne der SQL Server-Objekte werden wiederverwendet.



Migration Access zu SQL Server

Links

- Upsizing-Assistent
 - <http://support.microsoft.com/kb/294407/en-us>
 - <http://support.microsoft.com/kb/325017/en-us>
 - <http://support.microsoft.com/kb/330468/en-us>

- SQL Server 2005 Migrations-Assistent für Access
 - <http://www.microsoft.com/sql/solutions/migration/access/default.mspx>
 - <http://www.microsoft.com/sql/solutions/migration/access/videos.mspx>

- www.access-im-unternehmen.de
- www.sqlfaq.de/blog



Migration Access zu SQL Server

Fragen

- Jetzt

- Später
 - info@berndjungbluth.de

- Ansonsten
 - Vielen Dank für die Aufmerksamkeit
 - Gute Heimreise